Customer Training Workshop Traveo™ II Serial Memory Interface (SMIF)

Q4 2020

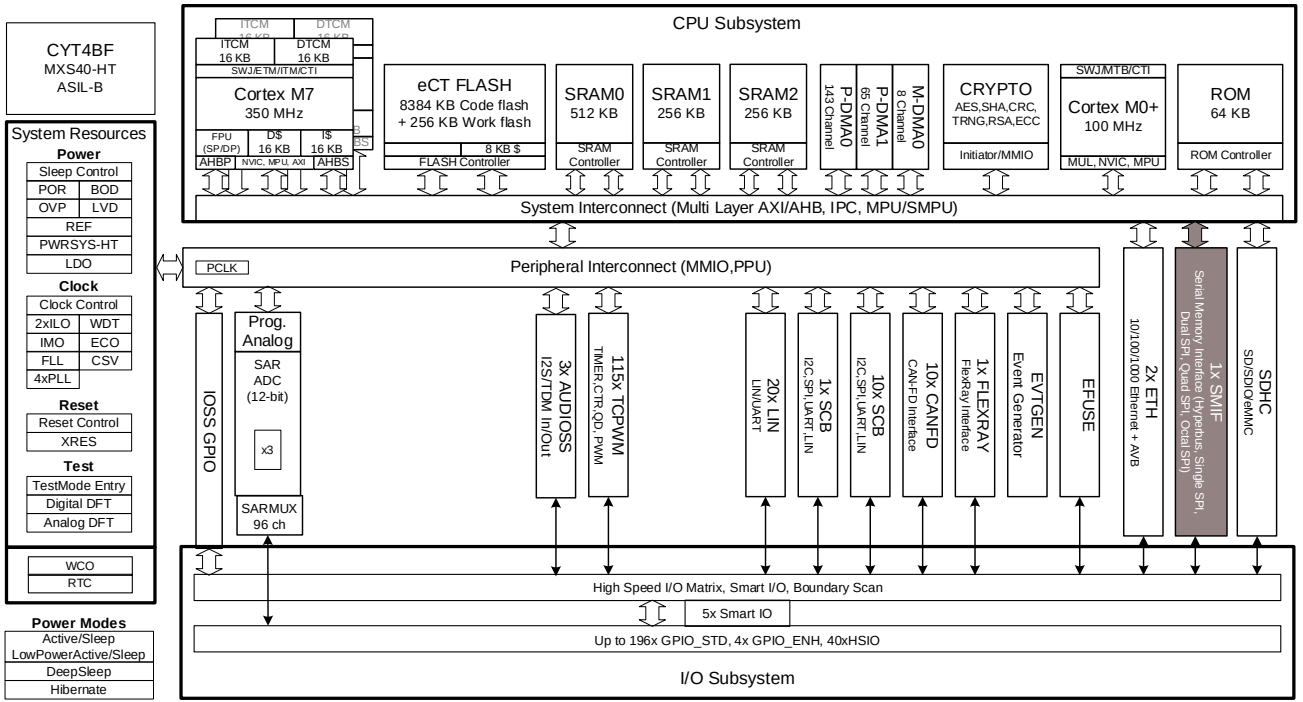
Target Products

› Target product list for this training material

Family Category	Series	Code Flash Memory Size
Traveo™ II Automotive Body Controller High	CYT3BB/4BB	Up to 4160KB
Traveo II Automotive Body Controller High	CYT4BF	Up to 8384KB
Traveo II Automotive Cluster	CYT3DL	Up to 4160KB
Traveo II Automotive Cluster	CYT4DN	Up to 6336KB

Introduction to Traveo II Body Controller High

› SMIF is located in the peripheral blocks

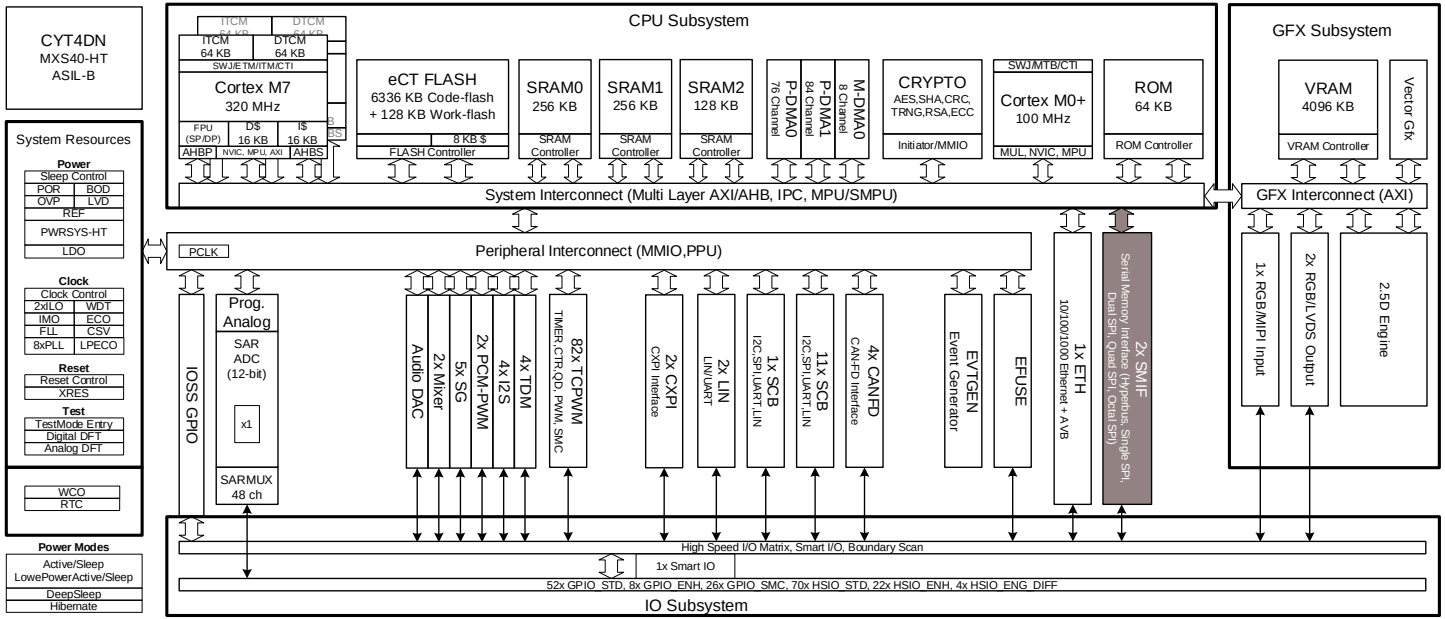


Hint Bar

Review TRM chapter 32 for additional details

Introduction to Traveo II Cluster

- SMIF is located in the peripheral blocks



Hint Bar

Review TRM chapter 32 for additional details

Serial Memory Interface (SMIF) Overview (1/2)

- SMIF provides an interface to memories with SPI and HyperBus IFs
- Features
  - SPI or HyperBus master functionality only
  - SPI protocol
    - Mode 0 only, with configurable Master-In Slave-Out (MISO) sampling timing
    - Single/Dual/Quad/Octal SPI transfer mode
    - Dual-quad SPI mode (two devices sharing one address range)
    - Single Data Rate (SDR) and Double Data Rate (DDR) transfer
  - Two operation modes
    - Memory Mapped I/O (MMIO) operation mode
    - eXecute-In-Place (XIP) operation mode
      - Read and write access
      - Automatic transition access from dedicated internal memory address to SPI protocol
      - On-the-fly encryption and decryption

Hint Bar

Review TRM section 32.1 for additional details

SPI Mode 0:
Data is driven on a falling edge of SPIHB_CLK

Data is captured on a rising edge of SPIHB_CLK

Serial Memory Interface (SMIF) Overview (2/2)

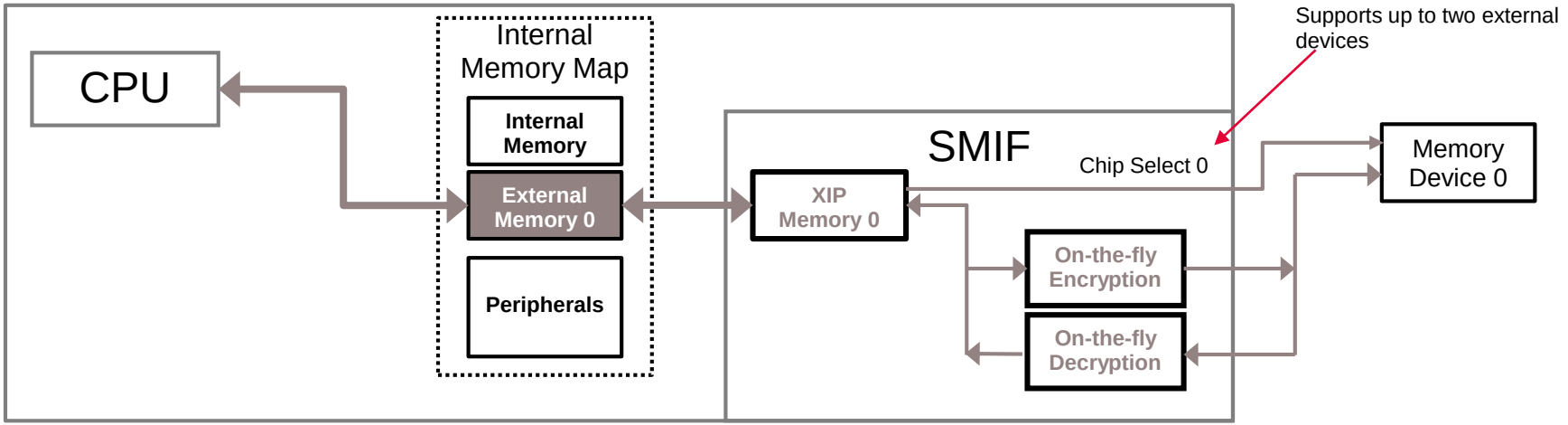
- Features
  - Memory Device
    - Device capacity in the range of 64KB to 4GB (memory size: 2^N)
      - (CYT4BF: 128MB area in XIP mode)
      - (CYT4DN: 512MB x2 area in XIP mode)
    - Support for configurable external device capacities
    - Up to two external memory devices sharing one address range (e.g. Dual-quad mode)
    - Up to four chip selects available
      - (CYT4BF: 2 pcs)
      - (CYT4DN: 2 pcs per channel)
  - Memory Interface Logic
    - Support stalling of transfers to address the back pressure on FIFOs
    - Support read-write-data-strobe (RWDS)
    - Support data signal connections between flexible external SPI memory devices
    - Support delay line and data learning pattern (DLP)-based data capture (CYT4DN)

Hint Bar

Review TRM section 32.1 for additional details

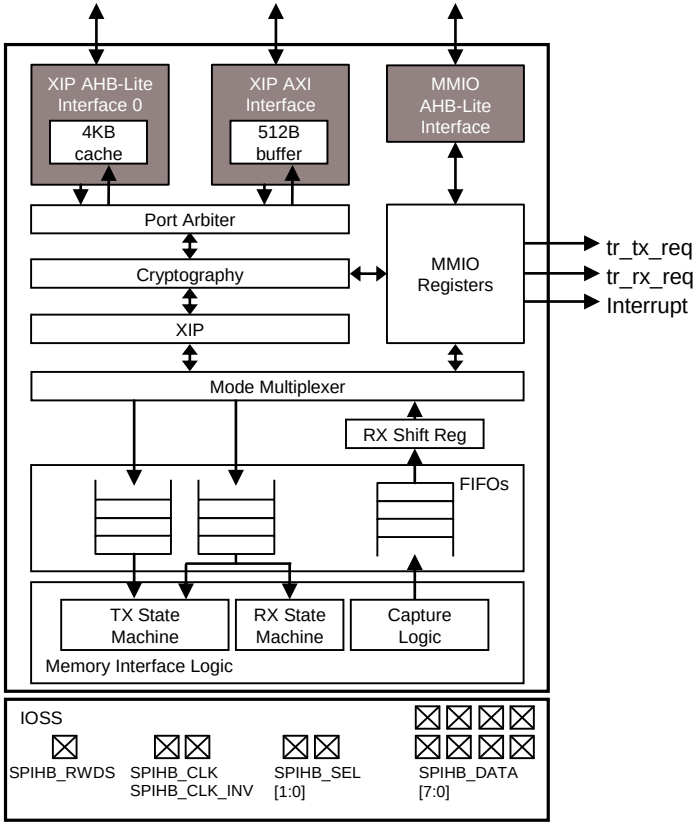
Example of Using SMIF

- External memory for code and data space
- Directly mapped to internal memory map (XIP operation mode)
- Encrypted/decrypted code or data in an external space



SMIF Block Diagram

- SMIF block components
  - Bus interface
    - MMIO AHB-Lite interface
    - XIP AXI interface
    - XIP AHB-Lite interface



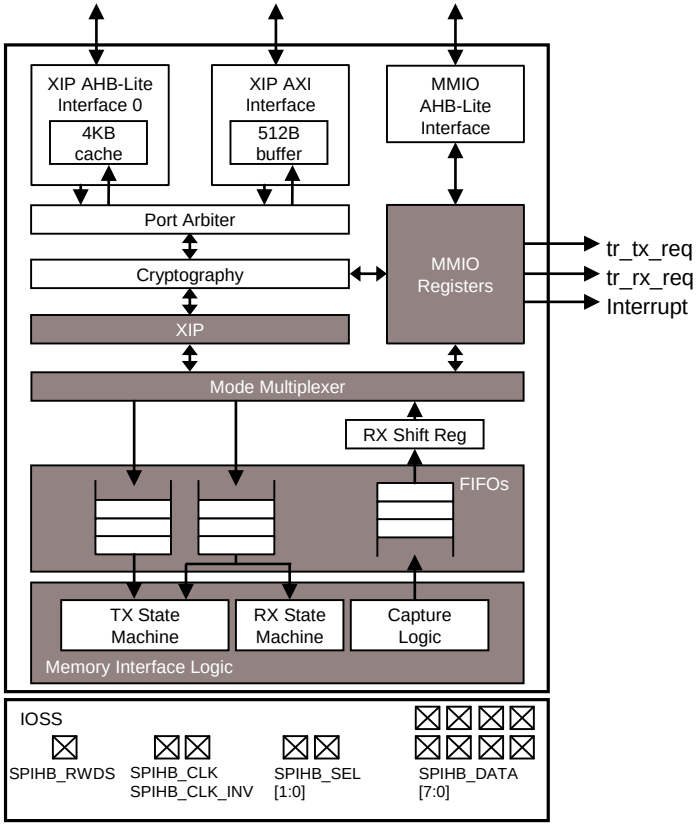
Bus Interface

- All bus interfaces provide access to external memories
- Any of the three bus interfaces can access external memories at any time
  - MMIO AHB-Lite interface
    - Access to the MMIO registers
    - Supports MMIO operation mode
  - XIP AXI interface
    - AXI interface for CPUSS fast domain[1]
    - Supports XIP operation mode
    - Fixed 512B buffer[2]
  - XIP AHB-Lite interface
    - AHB-Lite interface for CPUSS slow domain[3]
    - Supports XIP operation mode
    - 4KB read-only cache[3]
      - The “hit” read transfers are processed by the cache
      - Four-way set associative with an LRU replacement scheme

[1] The CPUSS fast domain component is CM7 CPUs.
[2] Note that the cache and the AXI interface buffer are not retained in DeepSleep power mode.
[3] The CPUSS slow domain components are CM0+ Crypto and P-DMA.

SMIF Block Diagram

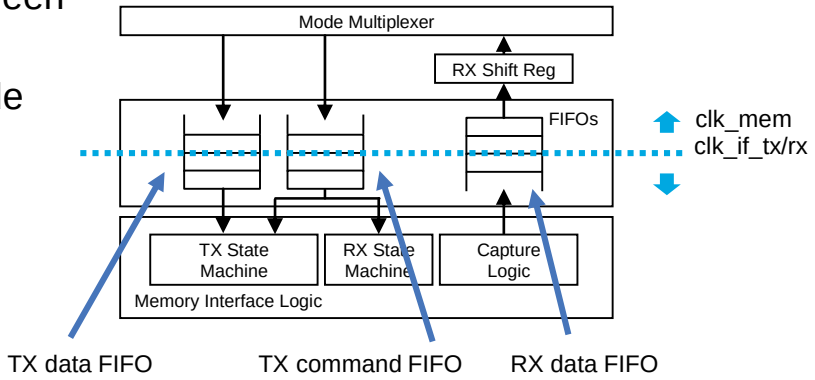
- SMIF block components
  - FIFOs
  - Two operation modes
    - MMIO operation mode
    - XIP operation mode
    - Mode is switched by XIP_MODE[1]
    - Both modes are mutually exclusive
  - Continuous transfer merging



[1] Refer to the Register TRM (SMIF_CTL) for additional details.

FIFOs

- SMIF has TX command FIFO, TX data FIFO, and RX data FIFO
  - Provide an asynchronous clock domain transfer between clk_mem and clk_if_tx/clk_if_rx
  - Software[1] controls the FIFOs in MMIO operation mode
  - MMIO registers[2] provide access to FIFOs



FIFO	Description
TX command FIFO	Transmit memory commands to the memory interface logic for memory transfer. Support five types of commands
TX data FIFO	Transmit write data to the memory interface logic
RX data FIFO	Receive read data from the memory interface logic

[1] In XIP operation mode, FIFOs are controlled by hardware.
[2] Refer to the Register TRM for additional details.

FIFOs (1/2)

- TX command FIFO supports five command types
  - TX command FIFO is controlled by the SMIF_TX_CMD_FIFO_WR.DATA27[26:0][1] register
  - DATA27[26:24] specifies the command and DATA[23:0] sets the command specification depending on command type

Command: DATA27[26:24]	Specification: DATA27[23:0]
TX: 0	- Width of the data transfer (single, dual, quad, or octal data transfer) - Data transfer mode (SDR or DDR) - External device select (multiple devices can be selected simultaneously) - The command is for the last phase of memory transfer - Transfer of 1-2 bytes A memory transfer must start with a TX command.
TX_COUNT: 1	This command is used to transmit data from TX data FIFO to external memories - Value of memory data to be transmitted - Width of the data transfer - Data transfer mode (SDR or DDR) - Specifies if this command is for the last phase of the memory transfer

[1] Refer to the Register TRM for additional details.

FIFOs (2/2)

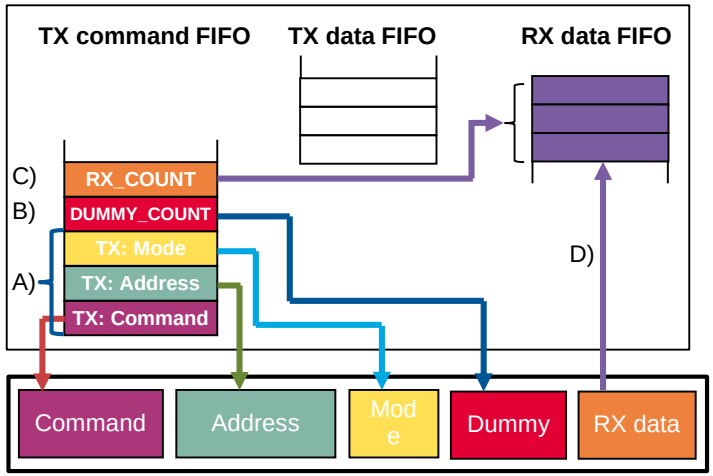
- TX command FIFO supports five command types
  - TX command FIFO is controlled by the SMIF_TX_CMD_FIFO_WR.DATA27[26:0][1] register
  - DATA27[26:24] specifies the command and DATA[23:0] sets the command specification depending on command type

Command: DATA27[26:24]	Specification: DATA27[23:0]
RX_COUNT: 2	This command is used by RX data FIFO to receive data from external memories - Value of the memory data received - Width of the data transfer - Data transfer mode (SDR or DDR) - Specifies if this command is for the last phase of the memory transfer
DUMMY_COUNT: 3	- Number of dummy cycles (used to implement turnaround time) - If the variable latency mode for HyperRAM is enabled causing double the number of dummy cycles, this command never constitutes the last phase of the memory transfer
DESELECT: 4	Finish a transfer and deselect the memory device This command always constitutes the last phase of the memory transfer

[1] Refer to the Register TRM for additional details.

Constructing a Read/Write Command

Read command



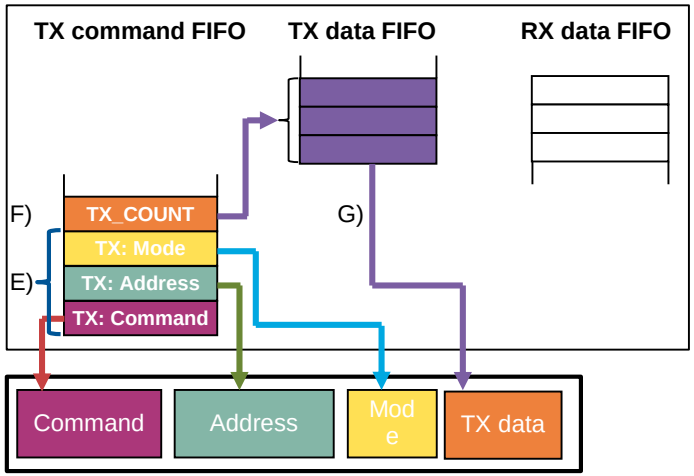
A) Set TX command FIFO in the order of Command, Address, and Mode using the “TX” command

B) Set dummy cycle to the TX command FIFO using the “DUMMY_COUNT” command

C) Set number of reception data to the TX command FIFO using the “RX_COUNT” command

D) Receive the number of data set by RX_COUNT in the RX data FIFO

Write command



E) Set TX command FIFO in the order of Command, Address, and Mode using the “TX” command

F) Set number of transmission data to the TX command FIFO using the “TX_COUNT” command

G) Transmit the number of data set by TX_COUNT from the TX data FIFO

Hint Bar

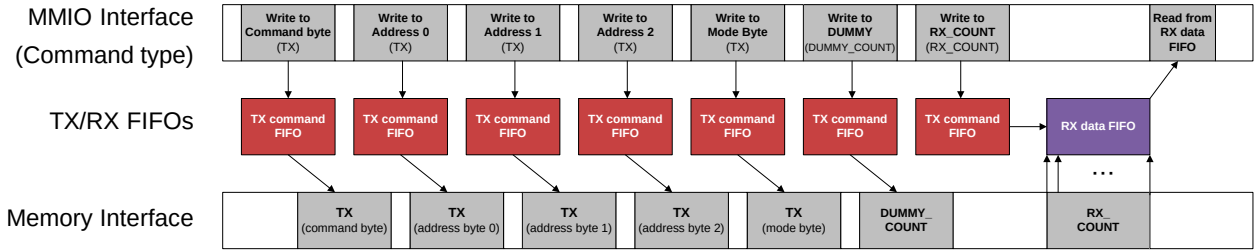
Review TRM section 32.1 for additional details

Mode:
It can eliminate the overhead of SIO instructions when repeating the same type of read command

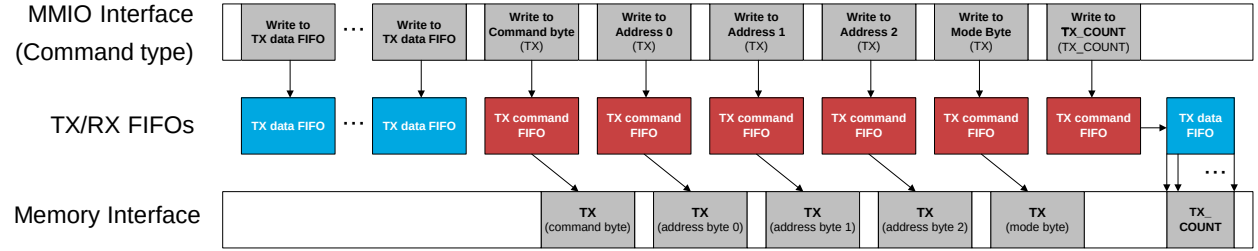
Operation Mode

- MMIO Operation Mode
  - Active by writing “0” to XIP_MODE1
  - Supports access through software using FIFOs
  - Provides the flexibility to implement any SPI device transfer

Data Read example



Data Write example



[1] XIP_MODE default value is “0” in the MMIO mode.

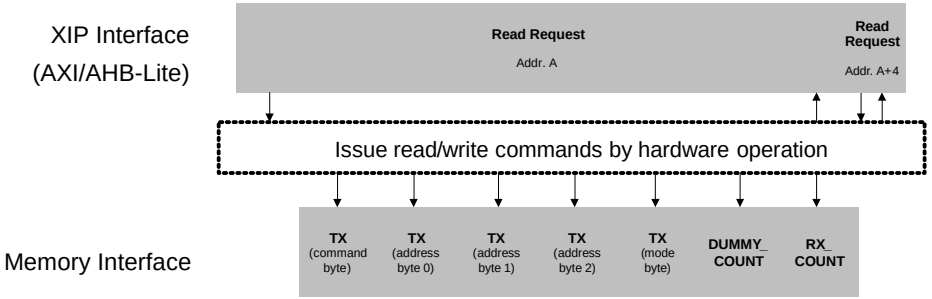
Hint Bar

The software must ensure that it generates correct memory transfers

Operation Mode

- XIP Operation Mode
  - Allows mapping each external memory into an internal memory address space
  - Active by writing “1” to XIP_MODE
  - Hardware generates memory transfers automatically without software intervention
  - Supports read and write access
  - FIFOs are controlled by hardware
  - Read/write commands are specified by MMIO registers[1] for each device

Data Read example



Hint Bar

Read/write commands specification includes:

Presence and value of SPI or HyperBus command byte

Number of address bytes

Presence and value of the mode byte

Number of dummy cycles

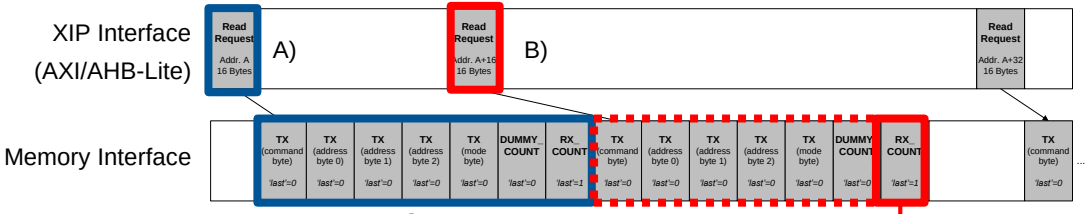
Data transfer widths and data transfer mode (SDR or DDR)

[1] Refer to the Register TRM (SMIF_DEVICE_RD/WR_CMD_CTL, _RD/WR_ADDR_CTL, _RD/WR_MODE_CTL, RD/WR_DUMMY_CTL, RD/WR_DATA_CTL) for additional details.

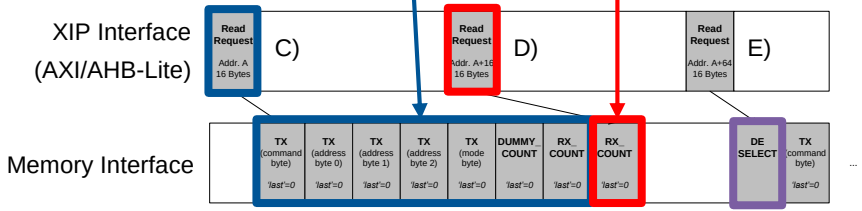
Continuous Transfer Merging

- The transfer can be merged to a single transfer at the memory interface[1] (MMIO and XIP modes)
  - Background: Longer transactions split on the AXI bus due to arbitration and crypto support
  - Avoid the overhead of multiple commands, addresses, modes, and dummy (latency) cycles

Read without continuous merging



Read with continuous merging



A) 16-byte read transfer from address A involves command, address, and dummy cycle

B) Read transfers from continuous addresses (A+16) also involves command, address, and dummy cycles

C) 16-byte read transfer from address A involves command, address, and dummy cycle

D) Read transfers from continuous addresses (A+16) do not require command, address, and dummy cycles

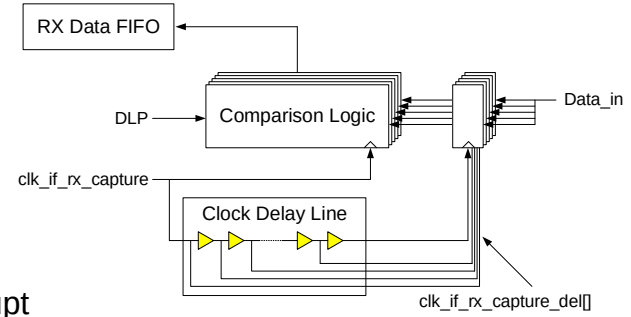
E) When a new transfer is requested or start address is not a continuation (A+64), a DESELECT command is generated
The next read transfer requires a command, address, and dummy cycle

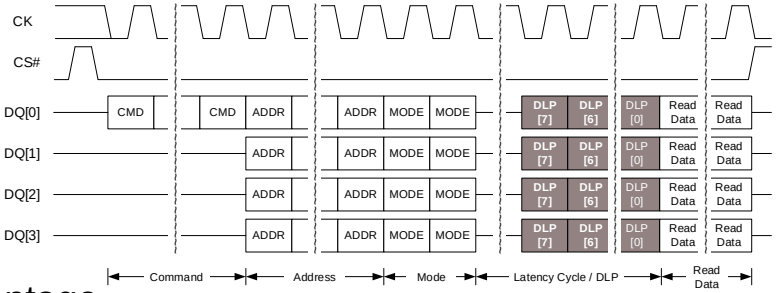
- Advantage
  - Improves performance of multiple continuous transfers

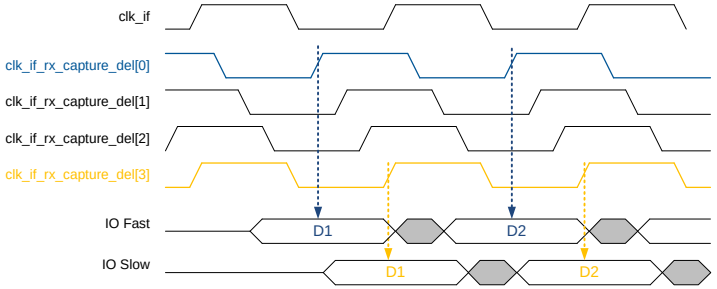
[1] Refer to the Register TRM (SMIF_CTL.MERGE_EN) for additional details.

Data Learning Pattern[1] (DLP)

- SMIF supports delay line and DLP-based data capture
  - Provides an optimal capture point within the data window
  - Captures input data with different clocks generated by taps of a delay line
  - Finds the delay tap for each data line by comparing the captured DLP with the expected one
  - Memory device provides a known DLP on every data I/O pin before outputting read data
  - No delay line tap resulting in a matching DLP, a data learning failed interrupt (in XIP/MMIO mode), and a bus error response (in XIP mode) being generated







- Advantage
  - Adjust signal delay easily due to wiring length, process, voltage, and temperature for each data line
  - Maximize read data throughput

[1] DLP is available in CYT4DN.

Bus Error Generation

› External memory accesses cause bus error under the following conditions

Interface	Precondition	Bus Error Factor
MMIO (AHB-Lite)	- SMIF_CTL.BLOCK = 0[1]	Software attempts to write an entry of a full TX command FIFO
		Software attempts to write more bytes than the available entries in the TX data FIFO
		Software attempts to read more bytes than the available entries in the RX data FIFO
XIP (AXI/AHB-Lite)	- SMIF_CTL.ENABLED = 0[1] - Write or read access	SMIF is disabled
	- SMIF_CTL.XIP_MODE = 0[1] - Write or read access	SMIF is not in XIP mode
	- XIP mode (SMIF_CTL.XIP_MODE = 1) - Write or read access	Transfer request is not in a memory region
	- XIP mode (SMIF_CTL.XIP_MODE = 1) - SMIF_DEVICE_CTL.WR_EN = 0[2]	Write transfer to a region that does not support writes
	- XIP mode (SMIF_CTL.XIP_MODE = 1) - Write access - Dual-quad SPI	Transfer address is not a multiple of 2
		Transfer size is not a multiple of 2
	- XIP mode (SMIF_CTL.XIP_MODE = 1) - Write access - Octal SPI DDR mode or HyperBus mode - Memory write byte masking is not supported	Transfer address is not a multiple of 2
		Transfer size is not a multiple of 2
	- SMIF_CTL.INT_CLOCK_DL_ENABLED = 1	No delay line tap resulting in a matching data learning pattern

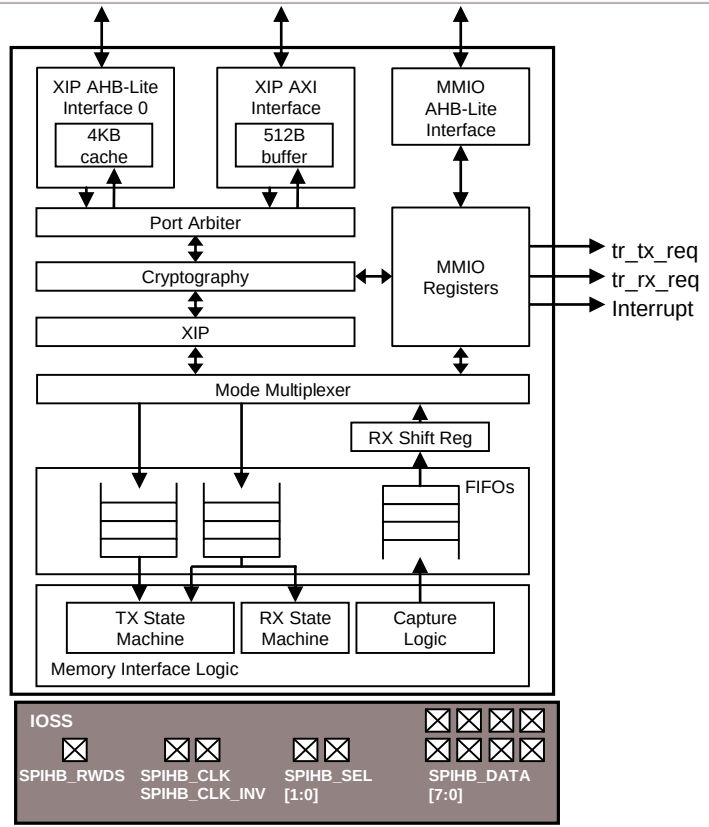
[1] Refer to the Register TRM (SMIF_CTL) for additional details.
[2] Refer to the Register TRM (SMIF_DEVICE_CTL) for additional details.

SMIF Block Diagram

- SMIF External Device Connection
  - Single, dual, quad, and octal data transfer
  - Maximum memory interface clock:
    - Depends on memory and I/O type

		Memory Type		
Products	**I/O Type**	**SDR**	**DDR**	**HyperBus**
CYT4BF	GPIO_STD[1]	32 MHz	32 MHz	32 MHz
	HSIO_STD[2]	100 MHz	100 MHz	100 MHz
CYT4DN	HSIO_ENH[3]	166 MHz	100 MHz	200 MHz

  - Can connect up to four devices
  - Independent SPIHB_CLK_INV outputs
    - Less latency, and no overlapping of SPIHB_CLK



[1] Supports standard automotive GPIO. Refer to the device datasheet for additional details.
[2] Supports high-speed I/O standard for high-speed peripherals. Refer to the device datasheet for additional details.
[3] Supports high-speed I/O enhanced for high-speed peripherals. Refer to the device datasheet for additional details.

External Device Connection

- SPI Connection
  - Support for single, dual, quad, and octal SPI protocol
  - Data signal type configuration in the MMIO register1
  - Can be set independently for each memory device (chip select)

Data Signal Connections

DATA_SEL[1:0]	Single SPI (Full Duplex)	Dual SPI (Half Duplex)	Quad SPI (Half Duplex)	Octal SPI (Half Duplex)
0	SPIHB_DATA [0] = SI SPIHB_DATA [1] = SO	SPIHB_DATA [0] = IO0 SPIHB_DATA [1] = IO1	SPIHB_DATA [0] = IO0 : SPIHB_DATA [3] = IO3	SPIHB_DATA [0] = IO0 : SPIHB_DATA [7] = IO7
1	SPIHB_DATA [2] = SI SPIHB_DATA [3] = SO	SPIHB_DATA [2] = IO0 SPIHB_DATA [3] = IO1	Illegal setting	Illegal setting
2	SPIHB_DATA [4] = SI SPIHB_DATA [5] = SO	SPIHB_DATA [4] = IO0 SPIHB_DATA [5] = IO1	SPIHB_DATA [4] = IO0 : SPIHB_DATA [7] = IO3	Illegal setting
3	SPIHB_DATA [6] = SI SPIHB_DATA [7] = SO	SPIHB_DATA [6] = IO0 SPIHB_DATA [7] = IO1	Illegal setting	Illegal setting

[1] Refer to the Register TRM (SMIF_DEVICE_CTL) for additional details.

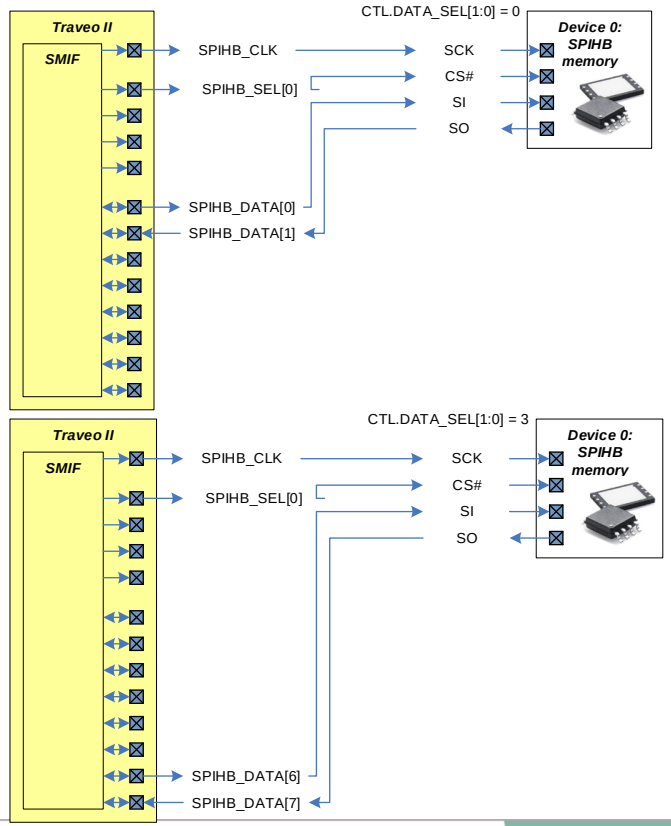
External Device Connection

- SPI Connection
  - The following diagrams show examples of a single SPI connection
- Features
  - Data signal connections are configured by DATA_SEL
  - Chip select signal setting is configured depending on the operation mode

Operation Mode	Setting Register	Chip Select	Note
MMIO	TX command FIFO DATA[23:20][1]	DATA[21]: SPIHB0_SEL[1] DATA[20]: SPIHB0_SEL[0]	
		DATA[23]: SPIHB1_SEL[1] DATA[22]: SPIHB1_SEL[0]	Supports CYT4DN
XIP	SMIF0_DEVICE0[2]	SPIHB0_SEL[0]	
	SMIF0_DEVICE1[2]	SPIHB0_SEL[1]	
	SMIF1_DEVICE0[2]	SPIHB1_SEL[0]	Supports CYT4DN
	SMIF1_DEVICE1[2]	SPIHB1_SEL[1]	

[1] Refer to the Register TRM (SMIF_TX_CMD_FIFO_WR) for additional details. Multi chip selects can be set at the same time in dual-quad mode.
[2] Refer to Register TRM (SMIF_DEVICE) for additional details.

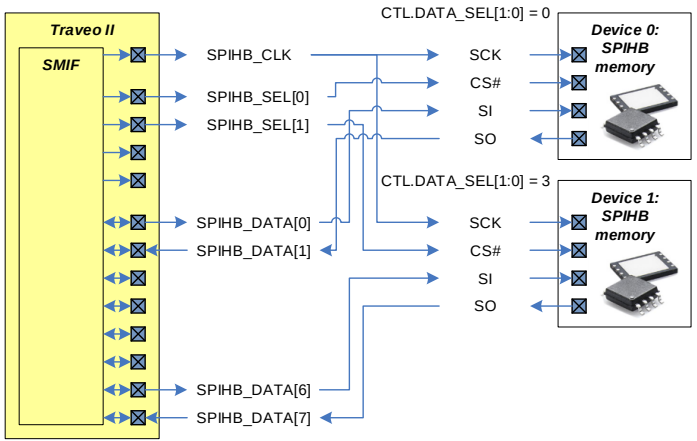
Single SPI Connection Examples



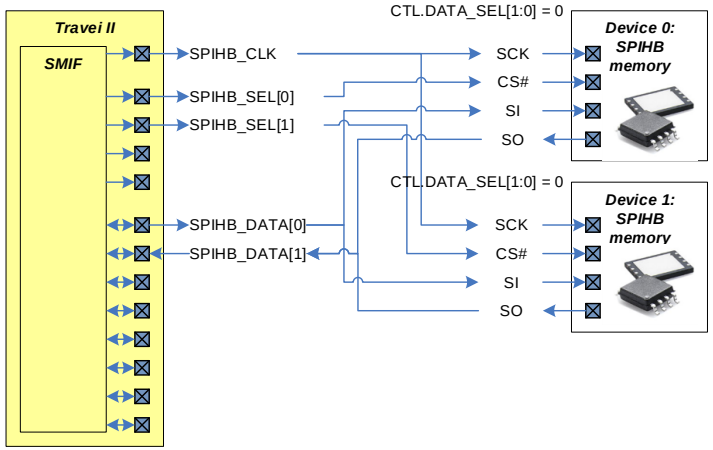
External Device Connection

- SPI Connection
  - The following diagrams provide examples of two single SPI device connections

Single SPI with Separate Data Signals



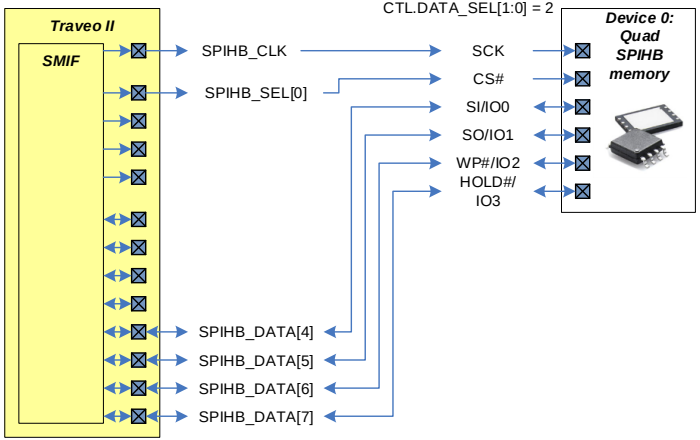
Single SPI with Shared Data Signals



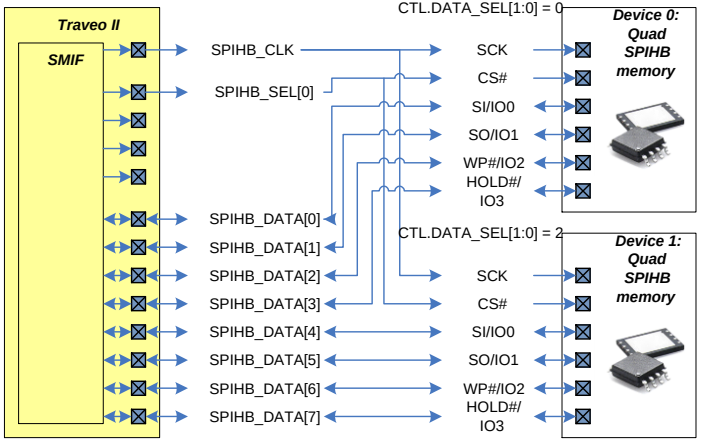
External Device Connection

- SPI Connection
  - The following diagrams show examples of quad and dual-quad SPI connections
- Features
  - Quad SPI can also use SPIHB_DATA [4:7] by the DATA_SEL configuration

Quad SPI Connection



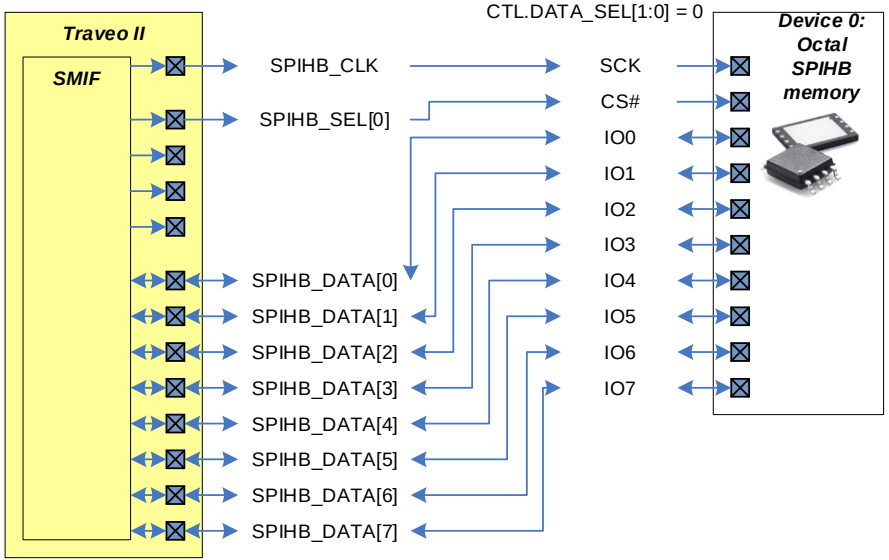
Dual-Quad SPI Connection (Pseudo Octal)



External Device Connection

- SPI Connection
  - The following diagram shows an example of an octal SPI connection
- Features
  - The only valid setting for DATA_SEL is “0”

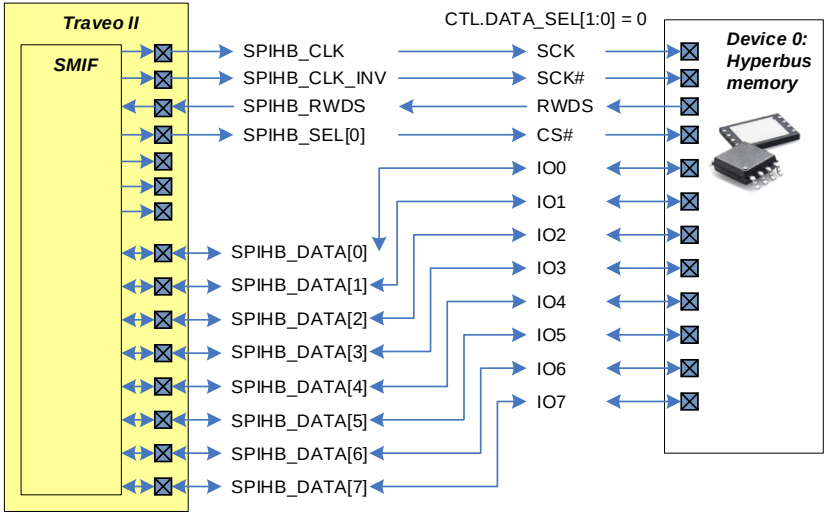
Octal SPI Connection



External Device Connection

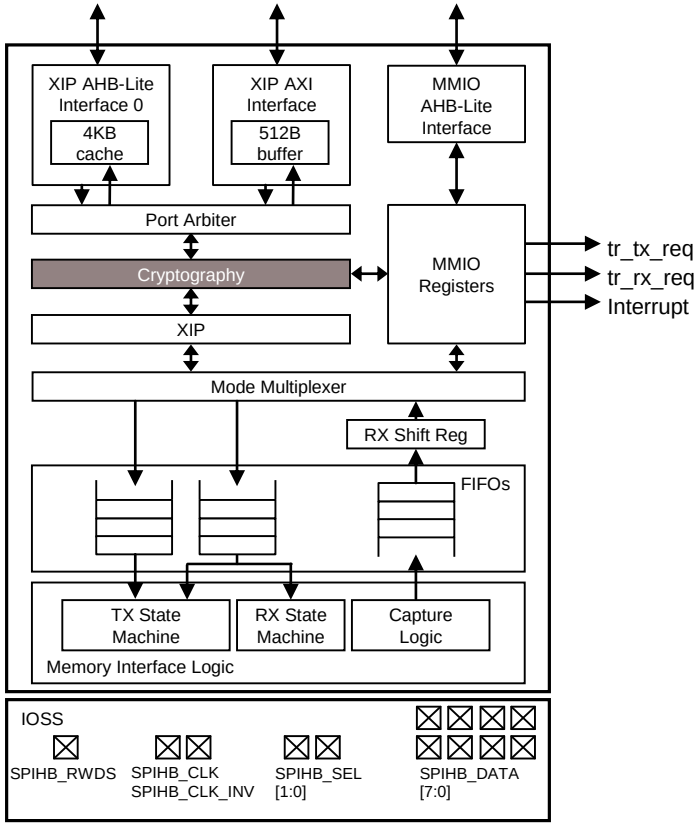
- HyperBus Connection
  - The following diagram shows an example of the HyperBus connection
- Features
  - Supports read-write-data-strobe (RWDS)
  - Only valid setting for DATA_SEL is “0”

HyperBus Connection



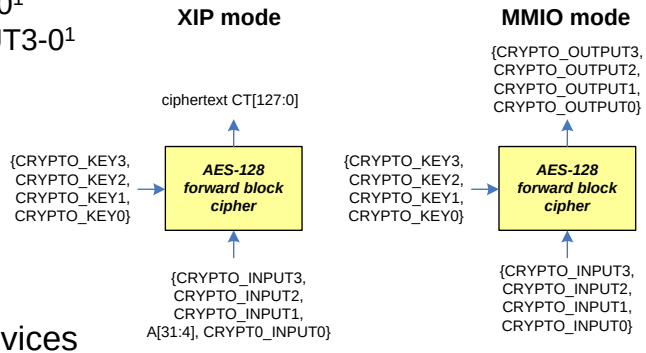
SMIF Block Diagram

- SMIF block component
  - Cryptography



Cryptography

- Read and write data transfers are available for encryption and decryption
  - Encryption and decryption are based on the AES-128 forward block cipher
    - The key is programmed into the SMIF_CRYPTO_KEY3-0 registers[1]
  - In MMIO mode
    - Supports offline encryption and decryption
    - Provides plaintext in SMIF_CRYPTO_INPUT3-0[1]
    - Provides ciphertext in SMIF_CRYPTO_OUTPUT3-0[1]
  - In XIP mode
    - Supports on-the-fly encryption for write data
    - Supports on-the-fly decryption for read data
    - Uses XIP address for plaintext and extends to SMIF_CRYPTO_INPUT3-0
- Advantage
  - Storing data encrypted in external memory devices prevents leakage of sensitive data



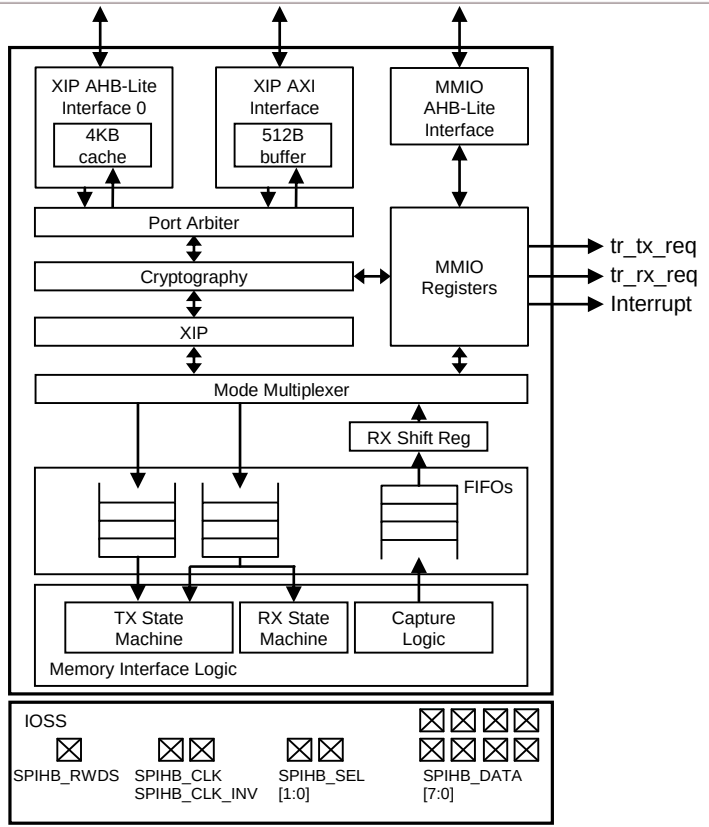
Hint Bar

SMIF_CRYPTO_KEY registers are write-only. Software read returns “0”.

[1] Refer to the Register TRM for additional details.

SMIF Block Diagram

- SMIF block components
  - Trigger and interrupt
    - tr_tx_req
    - tr_rx_req
    - Interrupt



Trigger and Interrupt

› SMIF has two triggers

Trigger	Condition
tr_tx_req	Active when the number of used TX data FIFO entries is smaller than or equal to the specified number[1]. Activated in MMIO operation mode
tr_rx_req	Active when the number of used RX data FIFO entries is greater than or equal to the specified number[2]. Activated in MMIO operation mode

[1] Refer to the Register TRM (SMIF_TX_DATA_MMIO_FIFO_CTL and SMIF_TX_DATA_MMIO_FIFO_STATUS) for additional details.
[2] Refer to the Register TRM (SMIF_RX_DATA_MMIO_FIFO_CTL and SMIF_RX_DATA_MMIO_FIFO_STATUS) for additional details.

Trigger and Interrupt (1/2)

› SMIF has one interrupt

Interrupt Cause	Condition
TR_TX_REQ	Active when the tr_tx_req trigger is activated. Activated in MMIO operation mode
TR_RX_REQ	Active when the tr_rx_req trigger is activated. Activated in MMIO operation mode
XIP_ALIGNMENT_ERROR[1]	- The XIP AHB-Lite/AXI bus transfer address is not a multiple of “2” or - The requested XIP AHB-Lite/AXI bus transfer size is not a multiple of “2” In the above conditions a) a write transfer is requested and b) Dual-Quad SPI mode is selected or Octal SPI DDR mode or HyperBus mode is selected without memory write byte masking Activated in XIP operation mode.
TX_CMD_FIFO_OVERFLOW	Write transfer to TX command FIFO[2] with not enough free entries available. Activated in MMIO operation mode

[1] Refer to the Register TRM (SMIF_INTR) for additional details.
[2] Refer to the Register TRM (SMIF_TX_CMD_FIFO_WR) for additional details.
[3] Refer to the Register TRM (SMIF_TX_DATA_FIFO_WR1,WR2,WR4) for additional details.
[4] Refer to the Register TRM (SMIF_RX_DATA_FIFO_RD1,RD2,RD4) for additional details.

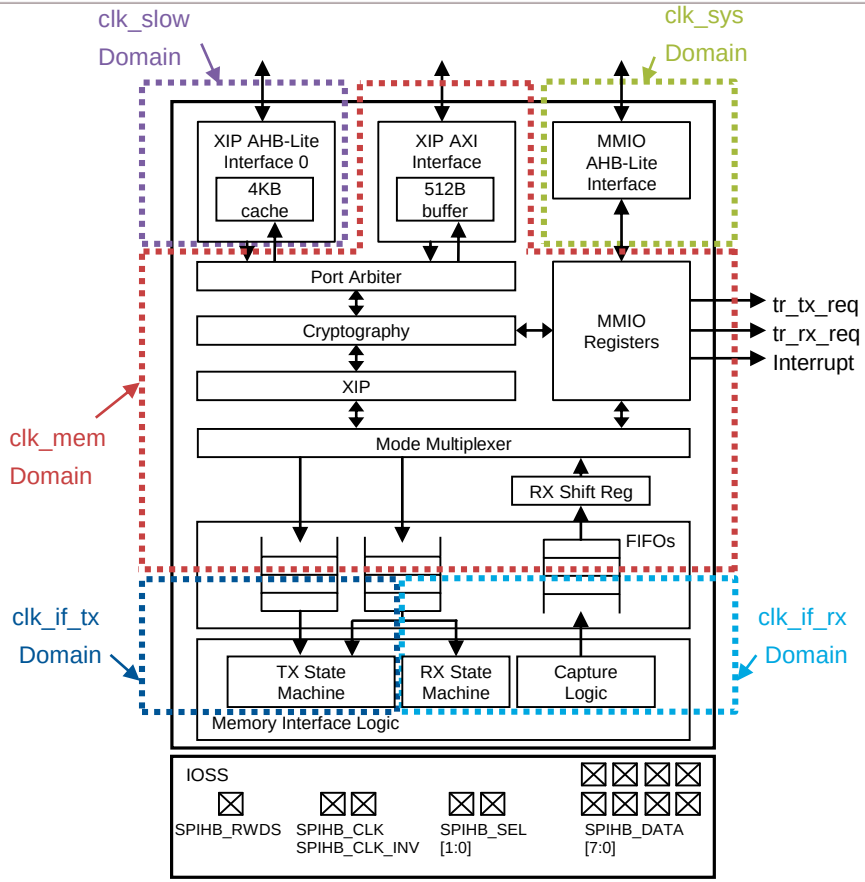
Trigger and Interrupt (2/2)

› SMIF has one interrupt

Interrupt Cause	Condition
TX_DATA_FIFO_OVERFLOW	Write transfer to TX data FIFO[3] with not enough free entries available. Activated in MMIO operation mode
RX_DATA_FIFO_OVERFLOW	Read transfer from RX data FIFO[4] with not enough entries available. Activated in MMIO operation mode
DL_FAIL	Data Learning Failed (no DLP match found on at least one of the input data lines when CTL.INT_CLOCK_DL_ENABLED = 1)
DL_WARNING	Data Learning Warning (for at least one input data line less then DLP.DL_WARNING_LEVEL delay line taps resulted in a correct DLP capturing when CTL.INT_CLOCK_DL_ENABLED = 1). This interrupt will be suppressed if DL_FAIL also occurs during the same DLP evaluation cycle
CRC_ERROR	CRC Error. A read transfer data CRC check failed
FS_STATUS_ERROR	Functional Safety Status Error. A read transfer Functional Safety Status check failed

SMIF Block Diagram

- SMIF block components
  - Clock domains
    - CLK_MEM
    - CLK_SLOW
    - CLK_SYS
    - clk_if_tx/rx



Clock

› The following clocks are used in SMIF

Source	Used Block	Function
clk_mem	SMIF Internal Block FIFOs	Used for the SMIF block; it is sourced from the mem clock The frequency range is up to 200 MHz
	AXI Master Interface	Used by the XIP AXI master interface; it is sourced from the mem clock The frequency range is up to 200 MHz
clk_slow	AHB Slave Interface	Used by the XIP AHB-Lite master interface; it is sourced from the slow clock The frequency range is up to 100 MHz
clk_sys	AHB Slave Interface	Used by the MMIO AHB-Lite slave interface; it is clocked by the PERI group clock The frequency range is up to 100 MHz
clk_if_tx/rx	Memory Interface FIFOs	Used for memory interface of transmit and receive; it is clocked by the CLK_HF6 clk_if_tx is the source clock for SPIHB_CLK

Hint Bar

Review TRM chapter 18 for additional details

Part of your life. Part of tomorrow.

Revision History

Revision	ECN	Submission Data	Description of Change
**	6399119	12/3/2018	Initial release
*A	6633414	7/22/2019	Change from mxsmif_ver2 to mxsmif_ver3 Updated slide 2, 6, 18, 19 Added slide 4, 17, 31
*B	7039074	12/03/2020	Updated page 2, 3, 4